<u>REMARKS</u>

Claims 1-27 are currently pending in the subject application and are presently under consideration. Favorable consideration of the subject patent application is respectfully requested in view of the comments herein.

I.   <u>Rejection of Claims 1-27 Under 35 U.S.C. §102(e)</u>

Claims 1-27 stand rejected under 35 U.S.C. §102(e) as being anticipated by Cohen *et al.* (US 6,324,543). Withdrawal of this rejection is respectfully requested for at least the following reasons. Cohen *et al.* fails to expressly or inherently describe each and every limitation set forth in the subject claims.

> A single prior art reference anticipates a patent claim only if it
> *expressly or inherently describes each and every limitation set*
> *forth in the patent claim. Trintec Industries, Inc. v. Top-U.S.A.*
> *Corp.*, 295 F.3d 1292, 63 USPQ2d 1597 (Fed. Cir. 2002); *See*
> *Verdegaal Bros. v. Union Oil Co. of California*, 814 F.2d 628, 631,
> 2 USPQ2d 1051, 1053 (Fed. Cir. 1987). The *identical invention*
> *must be shown in as complete detail as is contained in the ...*
> *claim. Richardson v. Suzuki Motor Co.*, 868 F.2d 1226, 9 USPQ2d
> 1913, 1920 (Fed. Cir. 1989) (emphasis added).

i.   *Claims 1, 14, 15, 23 and 27*

The invention as claimed relates to application developer coded extensible proxies that have access to method interception and remoting functionality and data, and in particular, to a system and method for interacting with an object where the system and method facilitate applications developers creating proxies, accessing method call interception functionality, retrieving information associated with a method that can be intercepted by the interception functionality and adapting and/or extending the functionality of object systems. The consequential beneficial result of the claimed invention being object systems that are more adaptable and/or extensible, and the mitigation of the inflexibility problems associated with conventional systems. Independent claims 1, 14, 15, 23 and 27, recite similar claims limitations: *a method call interceptor operable to intercept a method call to an object and to route the method call to a proxy, the method call interceptor accessible to application code; and an*

- 2 -

application code generic proxy operable to receive an intercepted method call, the application code generic proxy further operable to invoke a method on the object, to receive results from the object and to pass results to the entity that generated the intercepted method call. It is apparent that the invention as claimed utilizes a method call interceptor accessible to application code to intercept a method call to an object. Upon receipt of the method call by the method call interceptor, the method call interceptor routes the received method call to a proxy. Further the claimed invention utilizes an application code generic proxy to receive an intercepted method call, invoke a method on the object, receive results from the object and pass the results to the entity that generated the intercepted method call. Cohen *et al.* fails to expressly or inherently teach these exemplary limitations of the subject claims.

Cohen *et al.* discloses a method and system to allow programs to become dynamically reconfigurable without programmer intervention, *viz.*, the programs can be dynamically distributed among multiple computers within a computer network without modification to the source code of the programs running on the system. Further, Cohen *et al.* allows an administrator of the system to specify conditions under which reconfiguration is to occur without modification to the source text of the program to be dynamically reconfigured. *See*, Abstract. The Final Office Action dated August 26, 2004, indicates that Cohen *et al.* teaches *a method call interceptor operable to intercept a method call to an object and to route the method call to a proxy.* The Examiner directs applicants' representative's attention to col. 2, lines 1-5; and col. 3, lines 57-63, proxy B', and suggests that substantiation for these particular limitations may be found therein. Col. 2, lines 1-5, states: "From the bytecode files, the DOD system generates local and remote "proxies" for the object. The proxies intercept method calls to and from the object, and route the calls to the object. When the object is local (it has not been migrated), the local proxies routes calls directly to the object". It is apparent from the foregoing that Cohen *et al.*, generates *proxies for the object* and that the *proxies intercept the method calls.* The subject claims on the other hand utilize a *method call interceptor*, as distinct from a *proxy for an object*, to intercept method calls and route the method calls to *a proxy.* Thus, the invention as claimed in distinguishable on this ground.

- 3 -

The subject claims further recite that the *method call interceptor* is *accessible to application code*. The Examiner asserts that Cohen *et al.* teaches this limitation at col. 2, lines 1-10, which states:

> From the bytecode files, the DOD system generates local and remote "proxies" for the object. The proxies intercept method calls to and from the object, and route the calls to the object. When the object is local (it has not been migrated), the local proxy routes calls directly to the object; when the object is remote (it has been migrated), the local proxy routes calls to the remote proxy, which routes the calls to the object. A bytecode modification tool adjusts the names of the objects and references to the objects to ensure that name collisions do not occur between objects and their proxies.

However, it is inconceivable, since Cohen *et al.* fails to teach a *method call interceptor,* how Cohen *et al.* can teach a *method call interceptor accessible to application code.* Nevertheless, despite the indicated passage's failure to disclose a method call interceptor, applicants' representative posits that the foregoing passage discloses a bytecode modification tool that modifies bytecode files prior to the generation of *proxies* by the DOD system, as opposed to *a method call interceptor accessible to application code,* and thus the invention as claimed is clearly distinguishable with respect to Cohen *et al.*

In addition, the subject claims recite *an application code generic proxy operable to receive an intercepted method call, the application code generic proxy further operable to invoke a method on the object, to receive results from the object and to pass results to the entity that generated the intercepted method call.* The Final Office Action dated August 26, 2004, suggests that *an application code generic proxy operable to receive an intercepted method call* can be found at col. 2, lines 1-5; and col. 3, lines 57-63, proxy B". While applicants' representative agrees with the Examiner that the cited passages provide for the generation of *proxies,* the *proxies* that are generated by the DOD system of Cohen *et al.* however, *intercept method calls to and from the object, and route calls to the object. See* col. 2, lines 2-4. The subject claims on the other hand, recite *an application code generic proxy operable to <u>receive</u> an intercepted method call,* rather than a *proxy that intercepts method calls.* Thus, it is apparent that the *application code*

- 4 -

generic proxy receives an intercepted method call rather than *intercepting the method call*. Further, the proxy as recited in the subject claims is an *application code generic proxy*, not a *code-generated proxy* as is disclosed in Cohen *et al.* The difference between the two types of proxies is that an *application code generic proxy* is one that can be employed to proxy many types of objects where the type of the object being proxied is passed to a generic proxy during construction of the proxy, *e.g.*, the generic proxy employs the supplied type information to behave as an instance of the supplied type with a resulting one-to-many mapping possible between generic proxies and the types being proxied. A *generated proxy*, in contrast, has a one-to-one mapping with the types being proxied and thus though *generated proxies* are simpler to implement as the code is inserted into the proxy at compile time, *generated proxies* are less efficient and more inflexible when compared with *application code generic proxies*. Moreover, Cohen *et al.* indicates at col. 3, lines 57-63 that the proxies generated by the DOD are *code generated proxies*, rather than *application code generic proxies*, as it is specifically stated at col. 3, lines 61-63, that "once this configuration has been established, it cannot be altered at run-time." Thus it would seem under Cohen *et al.*'s conception that once the DOD generates a proxy, the proxy is limited to the one-to-one typing imposed by the programmer prior to compile time, which is in direct contrast to the *application code generic proxy* taught by the claimed invention wherein the proxy has a many-to-one correspondence with the types of objects received from the method call interceptor.

In view of at least the foregoing it is submitted Cohen *et al.* fails to disclose all the limitations set forth in independent claims 1, 14, 15, 23 and 27, and claims that depend there from, and that this rejection should be withdrawn.

ii.   *Claim 24*

Independent claim 24 recites *a data packet adapted to be transmitted between two or more computer processes, the data packet comprising: one or more identifier/value pairs, the identifier identifying the value associated with the identifier/value pair, and the value providing information associated with an intercepted method call on an object.* The Examiner in the instant Final Office Action states: "it is inherent in Cohen's disclosure that data packets transmitted between

- 5 -

computer processes comprise identifier/value pairs providing information associated with an intercepted method call." Final Office Action dated August 26, 2004, page 3. Applicants' representative avers to the contrary. Nowhere in Cohen *et al.* are identifier/value pairs utilized, let alone disclosed, thus the supposed inherency that the Examiner seeks to rely upon is manifestly absent. Further, applicants' representative contends that all that is apparent from Cohen *et al.*'s disclosure is that it provides a method for allowing programs to become dynamically reconfigurable without programmer intervention, such that programmers can specify the conditions under which reconfiguration of the programs can occur. Accordingly, it is submitted that since Cohen *et al.* fails to disclose any of the limitations recited in independent claim 24, and those claims that depend there from, that this rejection should be withdrawn.
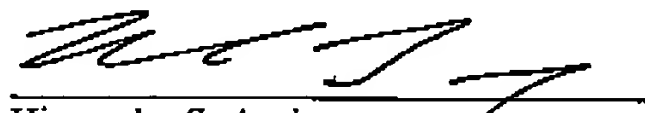
CONCLUSION

The present application is believed to be in condition for allowance in view of the above comments. A prompt action to such end is earnestly solicited.

In the event any fees are due in connection with this document, the Commissioner is authorized to charge those fees to Deposit Account No. 50-1063.

Should the Examiner believe a telephone interview would be helpful to expedite favorable prosecution, the Examiner is invited to contact applicants' undersigned representative at the telephone number below.

Respectfully submitted,

AMIN & TUROCY, LLP

Himanshu S. Amin
Reg. No. 40,894

AMIN & TUROCY, LLP
24TH Floor, National City Center
1900 E. 9TH Street
Cleveland, Ohio 44114
Telephone (216) 696-8730
Facsimile (216) 696-8731

- 6 -